**Maciej Oziębły**

**Nr indeksu: 184147**

Programowanie sieciowe

Sprawozdanie 3

Zakres sprawozdania:

* redukcja wymiarowości danych wejściowych (metoda PCA)
* rozpoznawanie liter za pomocą sieci RBF

## Testy numeryczne:

W celu przetestowania działania algorytmów wykorzystałem skrypt „testerka.m”, który był przeze mnie wykorzystywany na obu poprzednich ćwiczeniach (lekko zmodyfikowany):

letters = load\_letters\_definitions();

tries\_per\_letter = 50;

certainity\_level = 0.80;

avg\_tries\_per\_letter = zeros(size(letters,2),1);

[cut\_letters\_definitions, A\_matrix] = cut\_dimensions(load\_letters\_definitions(), certainity\_level);

weights = calc\_weights\_matrix(cut\_letters\_definitions);

for letter\_no = 1:size(letters,2)

tries = 0;

letter\_no

for i = 1:tries\_per\_letter

letter = letters(:, letter\_no);

random\_changes\_before\_fail = 0;

match\_failed = 0;

changed\_indexes = zeros(100,1);

while(match\_failed == 0 && random\_changes\_before\_fail ~= 100)

if(association\_recogniser\_optimized(letter, weights, A\_matrix) ~= mod((letter\_no-1), 35)+1)

%if(association\_recogniser(letter) ~= mod((letter\_no-1), 35)+1)

match\_failed = 1;

else

% generate random index not changed yet

rand\_index = randi(100);

while(max(ismember(changed\_indexes, rand\_index)) ~= 0)

rand\_index = randi(100);

end

changed\_indexes(length(changed\_indexes)+1) = rand\_index;

letter(rand\_index) = letter(rand\_index) \* -1;

random\_changes\_before\_fail = random\_changes\_before\_fail + 1;

end

end

tries = tries + random\_changes\_before\_fail;

end

avg\_tries\_per\_letter(letter\_no) = tries / tries\_per\_letter;

end

stem(avg\_tries\_per\_letter)

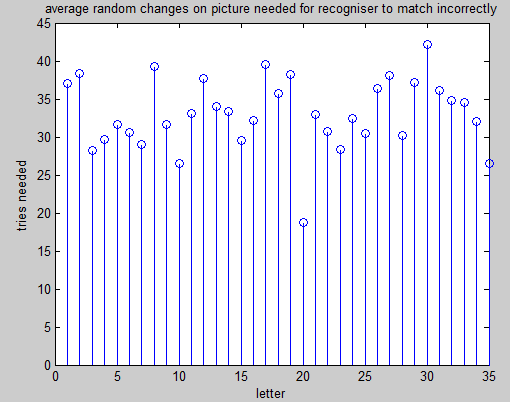
title('average random changes on picture needed for recogniser to match incorrectly')

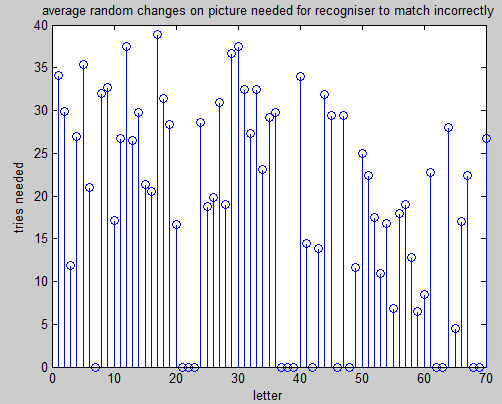
xlabel('letter')

ylabel('tries needed')

## Wyniki testów (redukcja wymiarów)

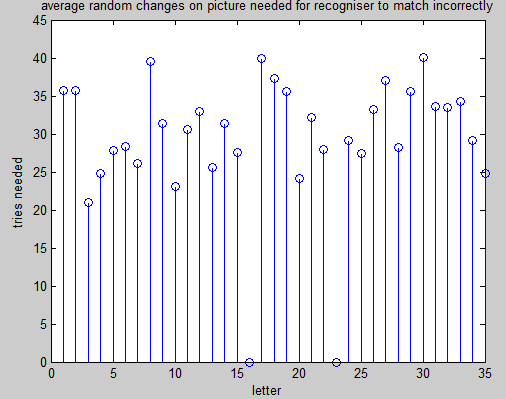
Dla stopnia pewności wynoszącego 1 (oczekiwane wyniki takie same jak przed zmniejszaniem wymiarowości):

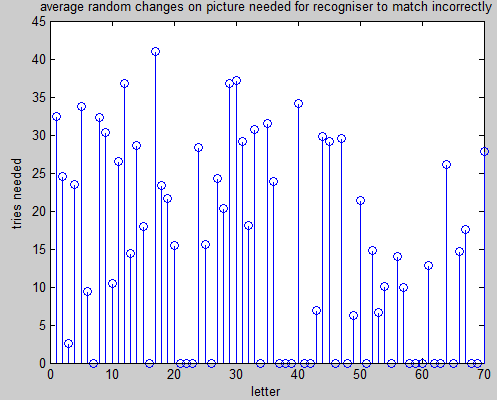




Wyniki spełniły moje oczekiwania – sieć radzi sobie z rozpoznawaniem tak samo jak przed redukcją wymiarów danych wejściowych.

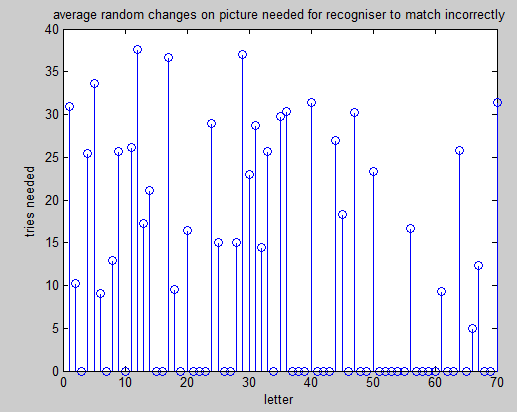
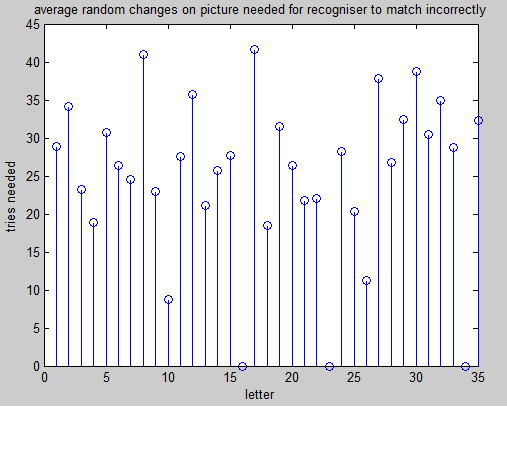
Po zmniejszeniu stopnia pewności do 0.95:





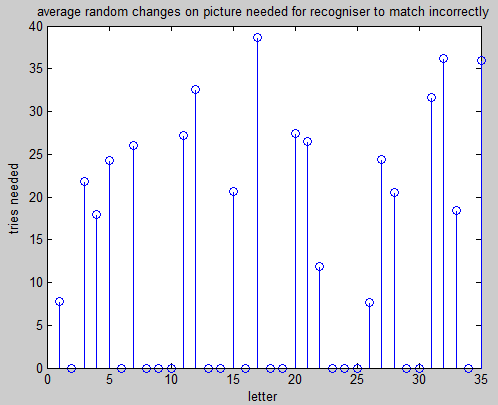
Przy obecności większej liczy wzorców widoczne jest znaczne pogorszenie wyników rozpoznawania liter. Dla małej ilości liter sieć nadal dobrze radzi sobie z rozpoznawaniem liter, jednak w 2 przypadkach od razu pojawiają się błędy.

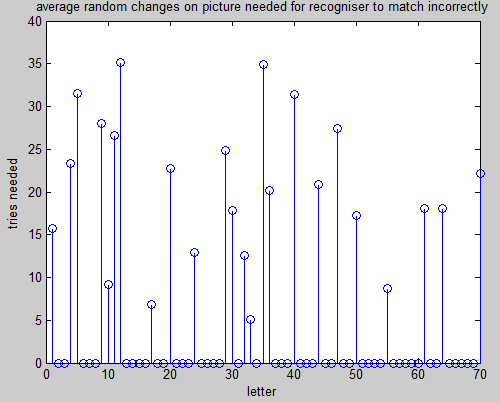
Dla poziomu pewności 0.9:



Sieć z dużą liczbą wzorców praktycznie przestała działać – dla małej liczby wzorców nie widać zmian.

Dla stopnia pewnośći 0.8:

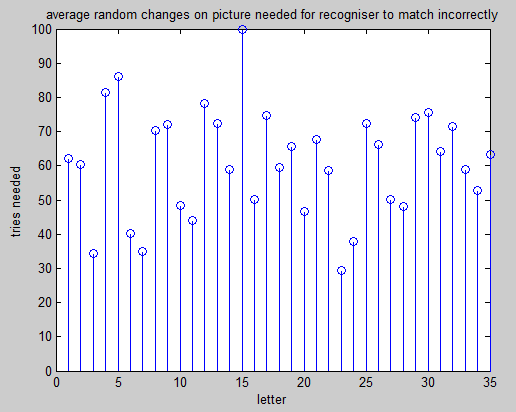




Przy stopniu pewnościu wynoszącym 0.8 sieć przestała radzić sobie z rozpoznawaniem liter zarówno dla małej, jak i dla dużej liczby wzorców.

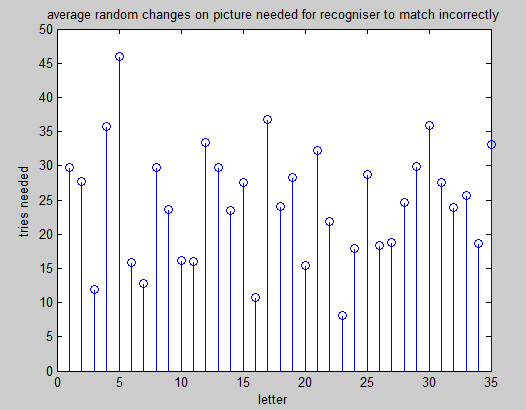
## Sieć RBF – wyniki testów:

Wykorzystałem sieć RBF do rozpoznawania wielkich liter (tak jak w poprzednim zadaniu):



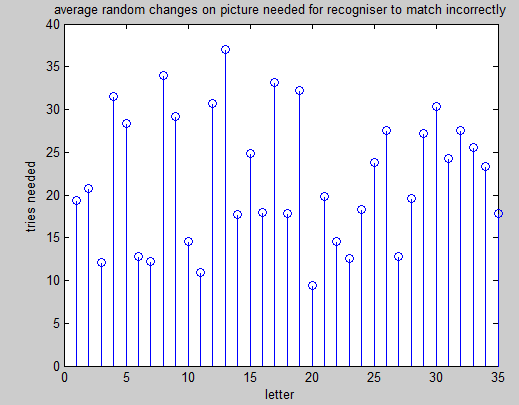
Okazuje się, że zastosowanie sieci RBF daje znacznie lepsze wyniki niż testowane wcześniej sieci asocjacyjne.

Po ogarniczeniu wymiarowości za pomocą metody PCA (przy stopniu pewności 0.95):

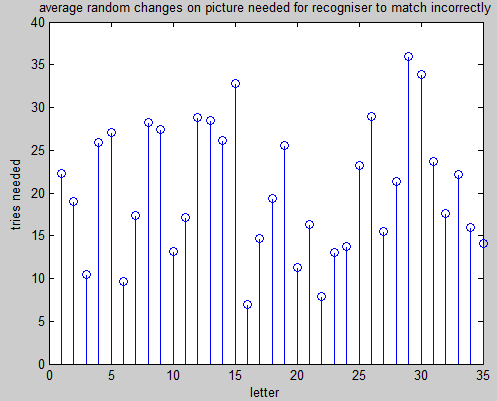


Widać tutaj znaczne pogorszenie wyników, ale algorytm zaczął działać znacznie szybciej.

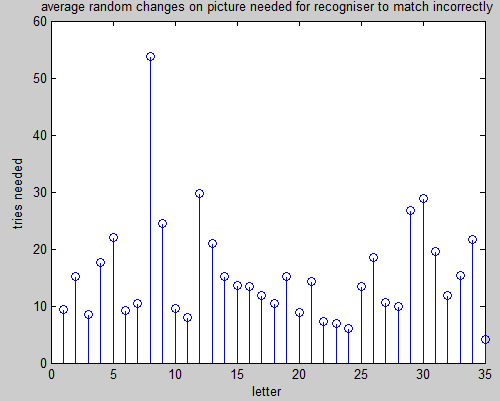
Po zmniejszeniu stopnia pewności do 0.9:



Po zmniejszeniu stopnia pewności do 0.85:



Po zmniejszeniu stopnia pewności do:



## Wnioski:

* ograniczenie liczby wymiarów danych wejściowych do sieci neuronowej metodą PCA pozwala na znaczne przyspieszenie działania algorytmu.
* korzystając z metody PCA optymalnie zmniejszamy liczbę danych (dla zadanego stopnoia pewności), ponieważ w tej metodzie maksymalizujemy wariancję danych po przekształceniu, co zapewnia zachowanie maksymalnie dużej liczby informacji.
* po zmniejszeniu stopnia pewności poniżej pewnego stopnia (w moim przypadku dla samych wielkich liter poniżej 0.9, dla małych i wielkich liter poniżej 0.95) sieć przstaje radzić sobie z rozpoznawaniem liter. Po nieznacznym zmniejszeniu stopnia pewności uzyskujemy duże przyspieszenie działania algorytmu, a działa on tylko nieznacznie gorzej.
* zastosowanie sieci RBF daje lepsze rezultaty niż wykorzystywane wcześniej liniowe i asocjacyjne sieci przy rozpoznawaniu liter
* skuteczność sieci RBF zmniejsza się bardzo wolno wraz ze zmniejszaniem wymiarowości danych wejściowych metodą PCA – nawet po zmniejszeniu stopnia pewności do 0.75 sieć RBF radziła sobie z rozpoznawaniem wszystkich liter do do pewneg – całkiem sporego stopnia zaszumienia.